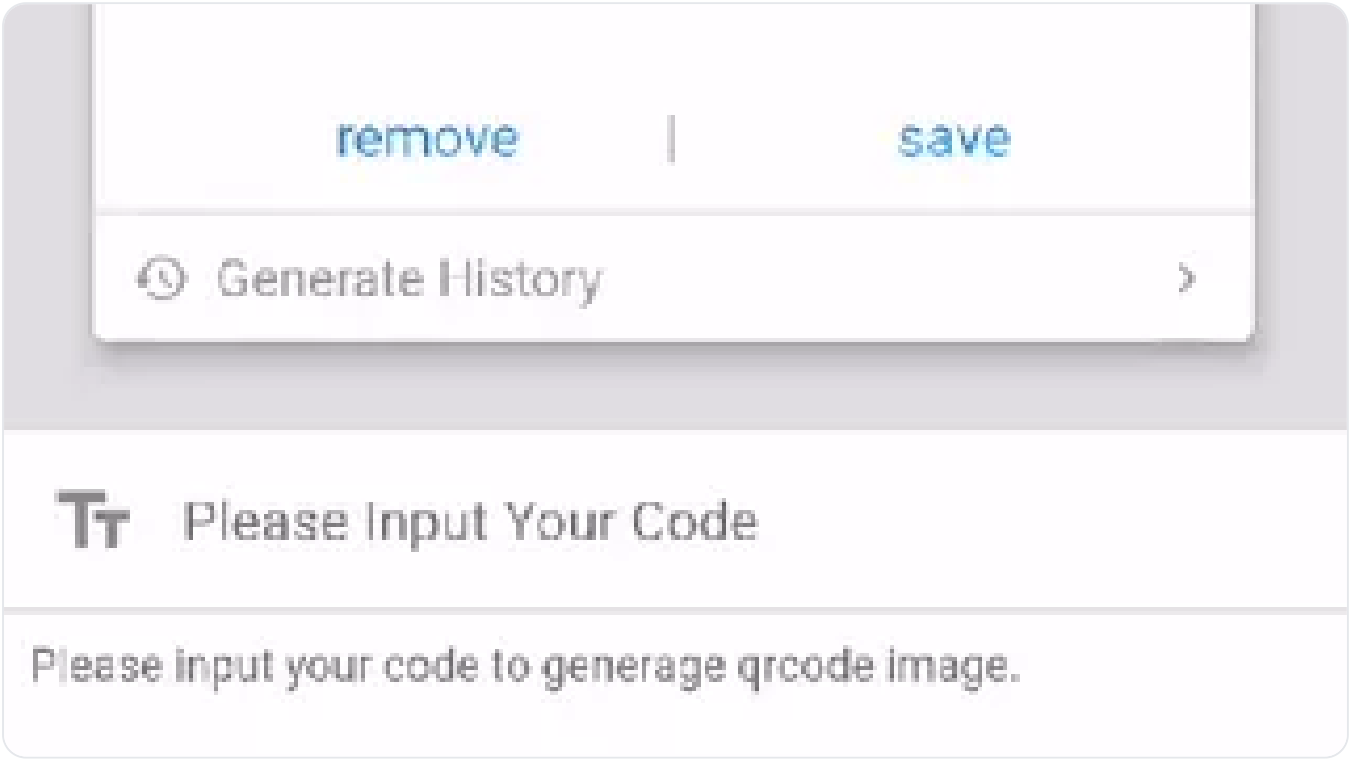
mobile_scanner

A Flutter barcode and QR scanner for Android, iOS, macOS, and web.



About

mobile_scanner is a Flutter barcode scanner for apps that need fast camera-based QR and barcode reading. It uses CameraX and ML Kit on Android, AVFoundation and Apple Vision on iOS and macOS, and ZXing on the web. The package is built for real-time detection, multiple barcode formats, and camera behavior that can be tuned for different scanning flows.

What is mobile_scanner

mobile_scanner is a Flutter barcode scanner package that lets you read QR codes and other barcode formats from a device camera. It is designed as a lightweight plugin for cross-platform apps and is published on pub.dev as version 7.4.0. The package supports

Android, iOS, macOS, and web, with no Linux or Windows support listed in the source material.

The implementation uses platform-specific scanning stacks: CameraX and ML Kit on Android, AVFoundation and Apple Vision on iOS and macOS, and ZXing on the web. That split matters because the scanning experience and available camera controls can differ by platform.

Key features

- Fast camera-based scanning for barcodes and QR codes.
- Support for multiple barcode formats rather than a single code type.
- Real-time detection so codes can be recognized as the camera feed updates.
- Customizable scanner behavior for apps that need a tighter scan area or different camera handling.
- Image analysis support on Android, iOS, and macOS.
- Optional image return support on Android, iOS, and macOS.
- Scan window support on Android, iOS, and macOS.
- Auto zoom support on Android only.
- Lens-type querying on Android and iOS, with facing filters for back or front cameras.
- Close-range lens selection helpers for devices that expose the right camera data.

Platform support is not identical across devices. For example, analyzeImage, returnImage, and scanWindow are not available on the web according to the package feature table. Lens controls are also more limited on macOS and web than on mobile.

How to use it

- Add the package to your Flutter project through pub.dev and import it in your app.
- Place the scanner widget or controller-driven scanner flow in the screen where you want to capture codes.
- Configure the camera behavior you need, such as a scan window, lens facing, or image-return settings.

If your app supports switching cameras, query supported lenses first so you only select hardware the device actually exposes.

A few of the package helpers are meant to make camera selection safer:

`getSupportedLenses()` returns the lens types available on the device.

Passing a `facing` filter narrows that list to one side of the camera hardware on Android and iOS.

`getBestCloseRangeScanningLens()` helps choose the lens that focuses closest, which can be useful for small codes held near the camera.

The source also shows that some examples exist in the package documentation, including basic usage, scan windows, and retrieving images from detected barcodes.

Who it is for

mobile_scanner is a good fit for Flutter developers building apps that need barcode reading without writing native scanner code from scratch. Typical uses include:

inventory and asset tracking apps

ticketing and check-in flows

retail or product lookup tools

internal utility apps that need QR code capture

apps that need a custom scan zone or live detection feedback

It is also relevant for developers who want one plugin that covers mobile and web, while still exposing platform-specific controls where they exist.

What to know before you use it

The package does not support Linux or Windows in the source data, so desktop coverage is limited to macOS only. Some features are platform-specific, and the web version has fewer camera and image-analysis capabilities than Android and iOS.

A few caveats are important:

`autoZoom` is supported on Android only.

On macOS and web, lens-type facing filters are ignored and all lenses may be returned.

`getBestCloseRangeScanningLens()` can return `null` if no camera is available for the requested facing direction.

The returned best lens is not guaranteed to be selectable, so you should confirm it against `getSupportedLenses()` before switching.

For device-specific availability, camera permissions, and platform quirks, your app should still rely on runtime checks and the official Flutter plugin documentation. The package gives you scanner building blocks, but it does not replace platform policy, permission handling, or hardware validation.

For teams that need a Flutter barcode scanner with cross-platform reach and configurable camera behavior, mobile_scanner is a practical starting point. If you need the most accurate feature details for a given device or OS version, verify them in the package docs and the official camera APIs before shipping.

FAQ

Is mobile_scanner free to use?

Which platforms does mobile_scanner support?

Does mobile_scanner work with QR codes only?

Can I scan QR codes and barcodes in real time?

Can I use mobile_scanner for both QR codes and barcodes?

Can I use mobile_scanner for QR codes and barcodes?

Can I control the scanner window or camera behavior?

Does mobile_scanner work on the web?

How do I choose the best camera lens for close-up scanning?

Does mobile_scanner work with Flutter on the web?

Can I customize the scanning area?

Who should use mobile_scanner?

When should I use getBestCloseRangeScanningLens()?

How do I choose the best camera lens for close-up scans?

What should I check before switching cameras?

Sponsored

Belay – Awa... Featured
Developer Tools

Stays awake while your agents work.

Free # Developer Tools

View ↗

Partly Featured
Finance

Partly is a mobile app that lets friends quickly scan receipts, split any bill fairl...

Free # Finance

View ↗

Visit website ↗

Pricing	Free
Category	Developer Tools
Platform	Android, iOS, macOS, Web
Launched	Aug 2026
Updated	today
Verified	5 days ago

TAGS

flutter # barcode-scanner # qr-code # camera # mobile-development # cross-platform # web-scan # web-support # pub-dev

© 2026 Dir Hub

mobile_scanner